**What Is Claimed Is:**

Sub A1

1. A method for verifying type safety of an application snapshot, the application snapshot including a state of an executing program that is moved from a first computing device to a second computing device across a network in order to continue execution on the second computing device, the method comprising:

receiving the application snapshot from the first computing device on the second computing device, wherein the application snapshot includes a subprogram, an operand stack, and a point of execution;

examining the application snapshot to identify the subprogram and the point of execution within the subprogram;

examining the subprogram to determine an expected structure of the operand stack at the point of execution;

validating that the state of the application snapshot on the second computing device is consistent with the expected structure of the operand stack; and

if the state of the application snapshot is validated as consistent with the expected structure of the operand stack, resuming execution of the application snapshot on the second computing device.

2. The method of claim 1, wherein examining the subprogram to determine the expected structure of the operand stack at the point of execution involves examining the subprogram with a code verifier, wherein the code verifier ensures that:

the subprogram does not cause the operand stack to overflow and underflow;

a use of a local variable does not violate type safety; and

an argument of an instruction is of an expected type.

Sub A1

3. The method of claim 1, wherein the operand stack contains at least one local variable, at least one argument that is passed as a parameter to the subprogram, and an offset to the point of execution within the subprogram.

4. The method of claim 2, wherein the expected structure of the operand stack includes a collective size of entries and the types of entries expected on the operand stack at the point of execution within the subprogram.

09575445.062800

5. The method of claim 1, further comprising restoring the state of an object within the application snapshot on the second computing device by changing a pointer from an address of the object on the first computing device to an address of the object on the second computing device.

6. The method of claim 4, wherein validating that the state of the application snapshot on the second computing device is consistent with the expected structure of the operand stack involves ensuring that the collective size of entries and the types of entries on the operand stack agree with the collective size of entries and the types of entries expected on the operand stack.

7. The method of claim 1, wherein resuming execution of the application snapshot involves restarting the subprogram at the point of execution within the second computing device.

8. A computer-readable storage medium storing instructions that when executed by a computer causes the computer to perform a method for

verifying type safety of an application snapshot, the application snapshot including a state of an executing program that is moved from a first computing device to a second computing device across a network in order to continue execution on the second computing device, the method comprising:

Sub A1

receiving the application snapshot from the first computing device on the second computing device, wherein the application snapshot includes a subprogram, an operand stack, and a point of execution;

examining the application snapshot to identify the subprogram and the point of execution within the subprogram;

examining the subprogram to determine an expected structure of the operand stack at the point of execution;

validating that the state of the application snapshot on the second computing device is consistent with the expected structure of the operand stack; and

if the state of the application snapshot is validated as consistent with the expected structure of the operand stack, resuming execution of the application snapshot on the second computing device.

9. The computer-readable storage medium of claim 8, wherein examining the subprogram to determine the expected structure of the operand stack at the point of execution involves examining the subprogram with a code verifier, wherein the code verifier ensures that:

the subprogram does not cause the operand stack to overflow and underflow;

a use of a local variable does not violate type safety; and

an argument of an instruction is of an expected type.

10. The computer-readable storage medium of claim 8, wherein the operand stack contains at least one local variable, at least one argument that is passed as a parameter to the subprogram, and an offset to the point of execution within the subprogram.

Sub A1

11. The computer-readable storage medium of claim 9, wherein the expected structure of the operand stack includes a collective size of entries and the types of entries expected on the operand stack at the point of execution within the subprogram.

09574115.062000

12. The computer-readable storage medium of claim 8, further comprising restoring the state of an object within the application snapshot on the second computing device by changing a pointer from an address of the object on the first computing device to an address of the object on the second computing device.

13. The computer-readable storage medium of claim 11, wherein validating that the state of the application snapshot on the second computing device is consistent with the expected structure of the operand stack involves ensuring that the collective size of entries and the types of entries on the operand stack agree with the collective size of entries and the types of entries expected on the operand stack.

14. The computer-readable storage medium of claim 8, wherein resuming execution of the application snapshot involves restarting the subprogram at the point of execution within the second computing device.

Sub A1

15. An apparatus that facilitates verifying type safety of an application snapshot, the application snapshot including a state of an executing program that is moved from a first computing device to a second computing device across a network in order to continue execution on the second computing device, comprising:

a receiving mechanism that is configured to receive the application snapshot from the first computing device on the second computing device, wherein the application snapshot includes a subprogram, an operand stack, and a point of execution;

an examination mechanism that is configured to examine the application snapshot to identify the subprogram and the point of execution within the subprogram wherein, the examination mechanism is configured to also examine the subprogram to determine an expected structure of the operand stack at the point of execution;

a validation mechanism that is configured to validate that the state of the application snapshot on the second computing device is consistent with the expected structure of the operand stack; and

an execution mechanism that is configured to resume execution of the application snapshot on the second computing device if the state of the application snapshot is validated as consistent with the expected structure of the operand stack.

16. The apparatus of claim 15, wherein the examination mechanism includes a code verifier, wherein the code verifier is configured to ensure that:

the subprogram does not cause the operand stack to overflow and underflow;

a use of a local variable does not violate type safety; and

an argument of an instruction is of an expected type.

17. The apparatus of claim 15, wherein the operand stack contains at least one local variable, at least one argument that is passed as a parameter to the subprogram, and an offset to the point of execution within the subprogram.

Sub A1

18. The apparatus of claim 16, wherein the expected structure of the operand stack includes a collective size of entries and the types of entries expected on the operand stack at the point of execution within the subprogram.

19. The apparatus of claim 15, further comprising an object restoring mechanism that is configured to restore the state of an object within the application snapshot on the second computing device by changing a pointer from an address of the object on the first computing device to an address of the object on the second computing device.

20. The apparatus of claim 18, wherein the validation mechanism is configured to ensure that the collective size of entries and the types of entries on the operand stack agree with the collective size of entries and the types of entries expected on the operand stack.

21. The apparatus of claim 15, wherein in resuming execution of the application snapshot, the execution mechanism is configured to restart the subprogram at the point of execution within the second computing device.